

Actuarial Science with

3. nonlife insurance (a quick overview)

Arthur Charpentier

joint work with **Christophe Dutang** & **Vincent Goulet**
and **Markus Gesmann's** `ChainLadder` package



MEIELISALP 2012 CONFERENCE, JUNE

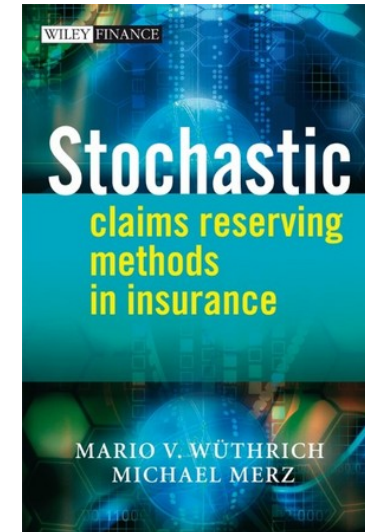
6th R/Rmetrics Meielisalp Workshop & Summer School
on Computational Finance and Financial Engineering

Some (standard) references

Wüthrich, M. & Merz, M. (2008)

Stochastic Claims Reserving Methods in Insurance

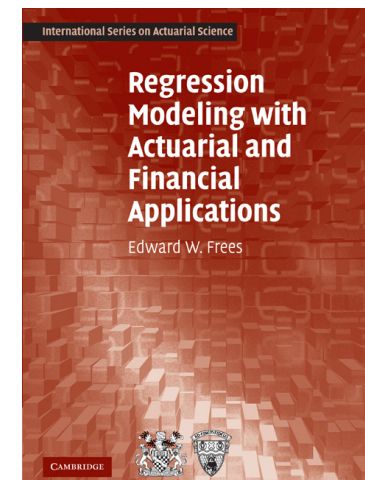
Wiley Finance Series



Frees, J.E. (2007)

*Regression Modeling with Actuarial
and Financial Applications*

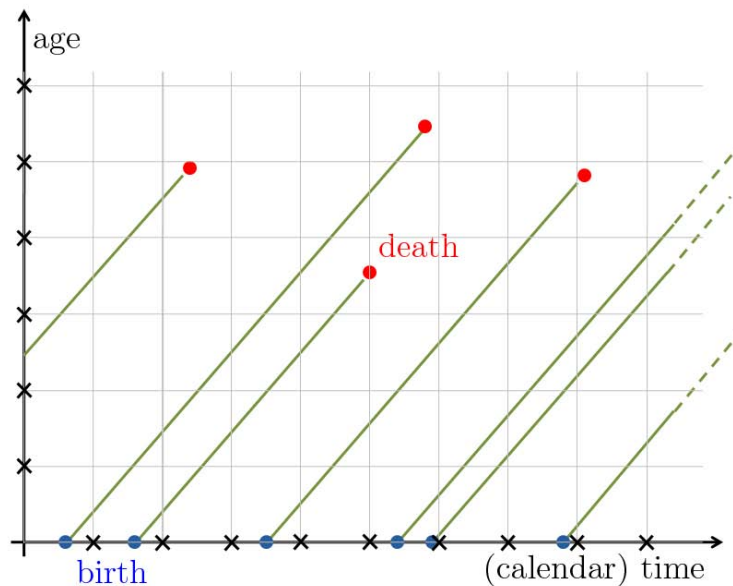
Cambridge University Press



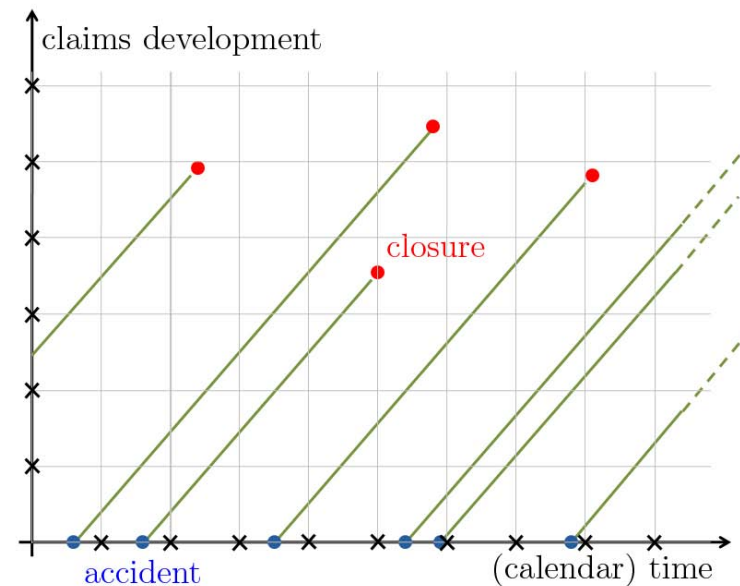
Lexis diagram in insurance

Lexis diagrams have been designed to visualize dynamics of life among several individuals, but can be used also to follow claims' life dynamics, from the occurrence until closure,

in **life** insurance



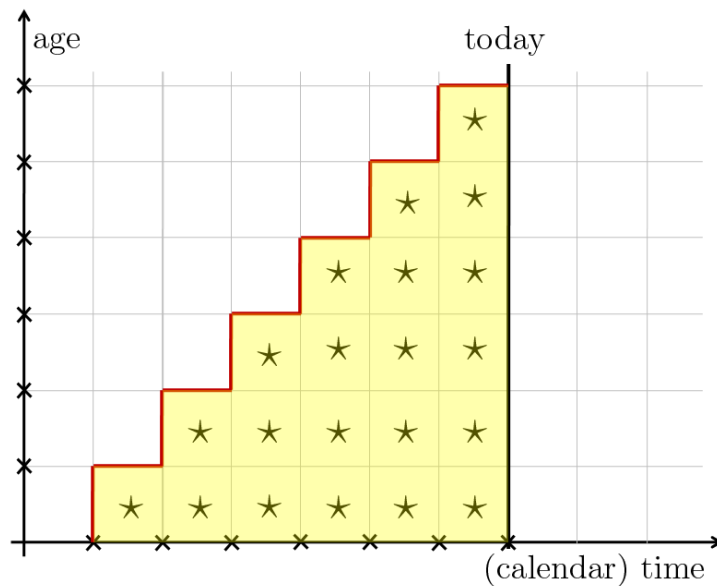
in **nonlife** insurance



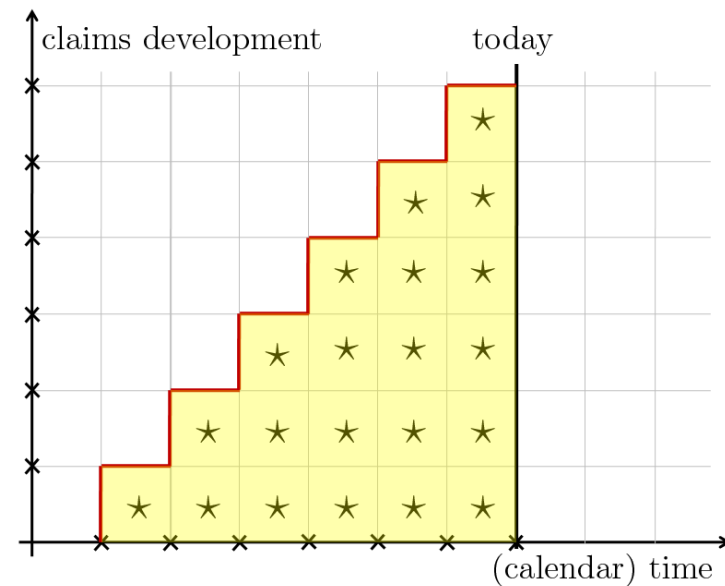
Lexis diagram in insurance

but usually we do not work on **continuous time** individual observations (individuals or claims) : we summarized information **per year**

in **life** insurance



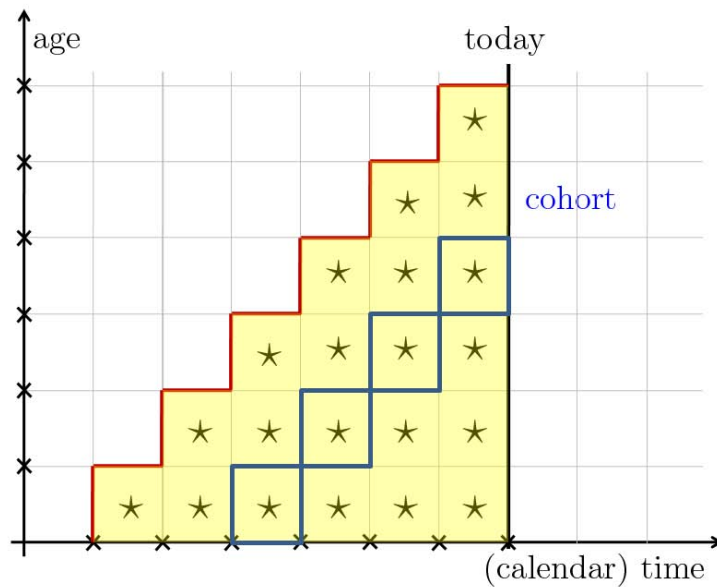
in **nonlife** insurance



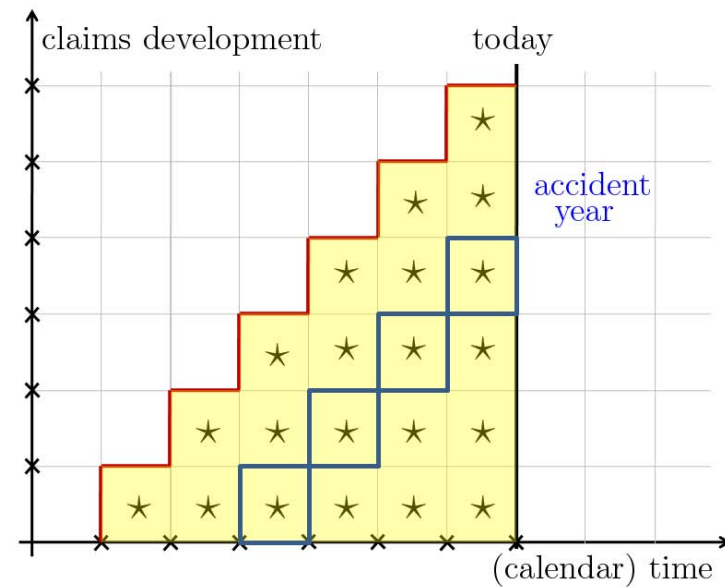
Lexis diagram in insurance

individual lives or claims can also be followed looking at **diagonals**,

in **life** insurance



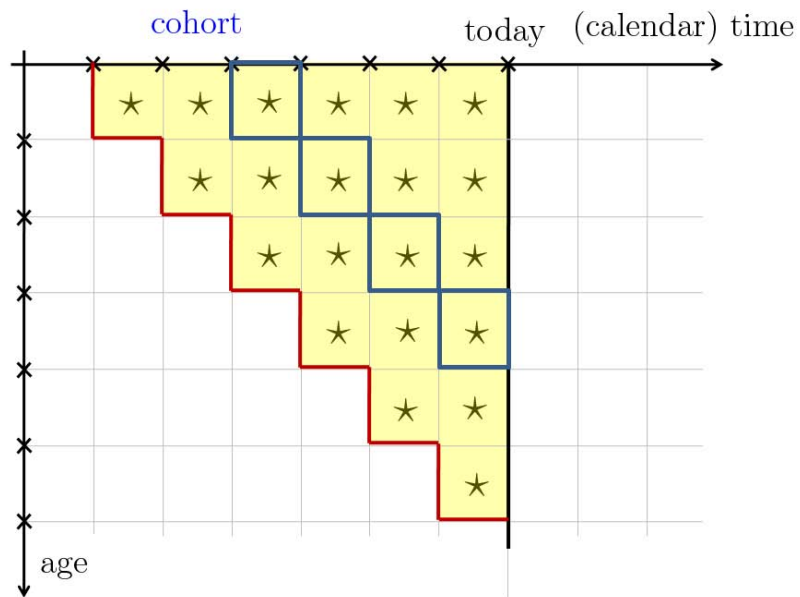
in **nonlife** insurance



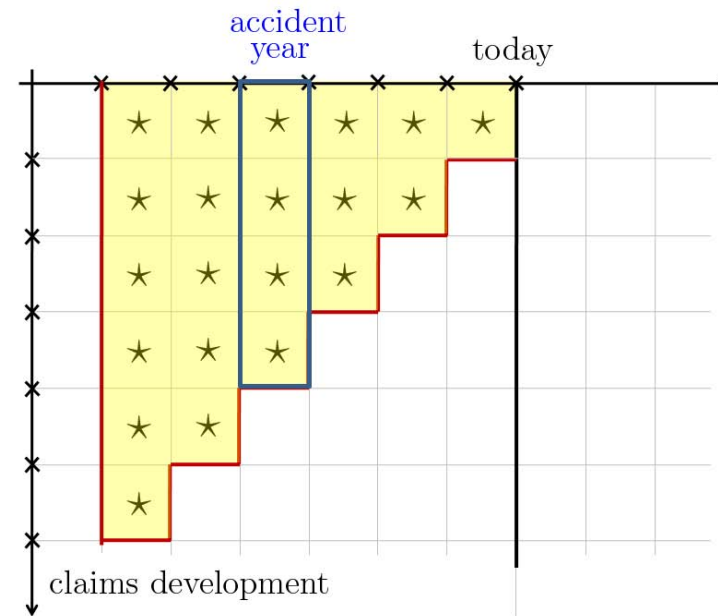
Lexis diagram in insurance

and usually, in nonlife insurance, instead of looking at (calendar) time, we follow observations per year of birth, or year of occurrence

in **life** insurance



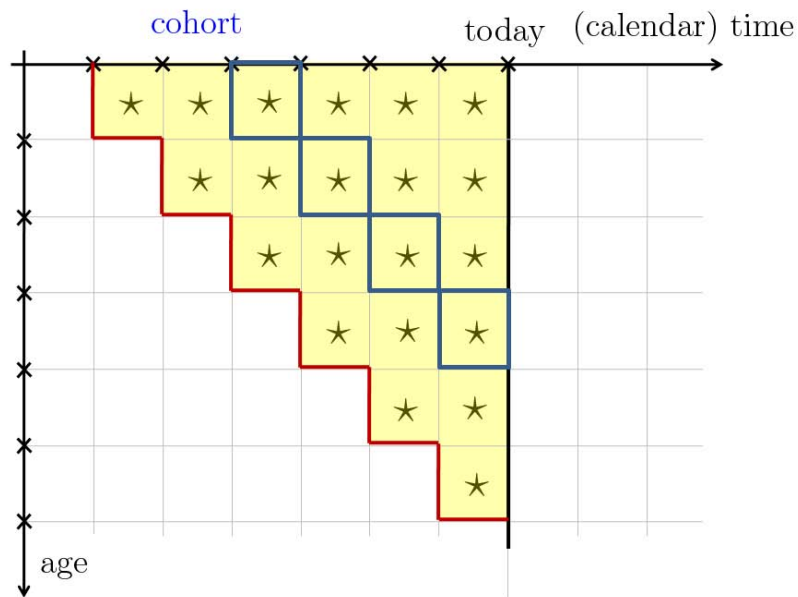
in **nonlife** insurance



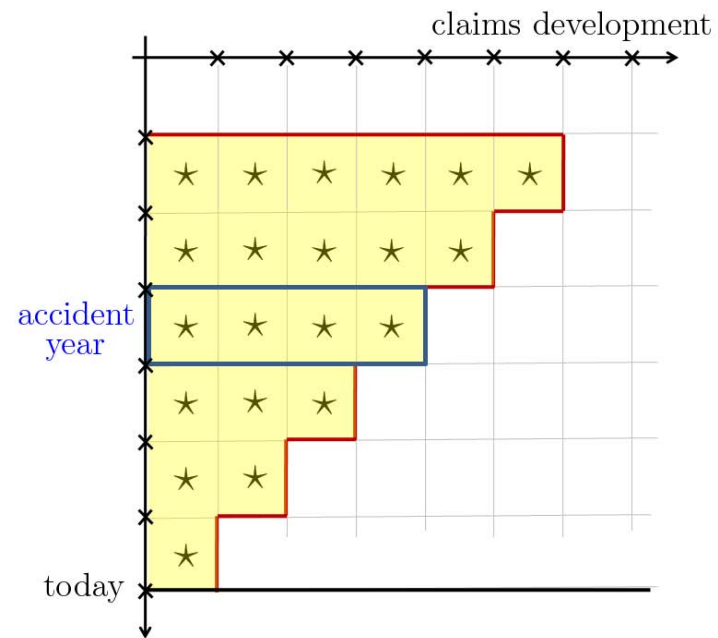
Lexis diagram in insurance

and finally, recall that in standard models in nonlife insurance, we look at the **transposed** triangle

in **life** insurance



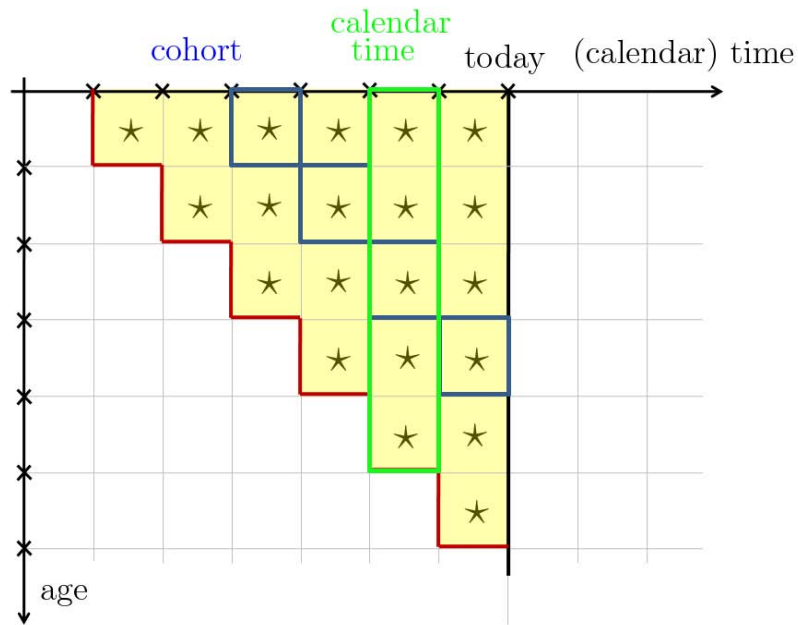
in **nonlife** insurance



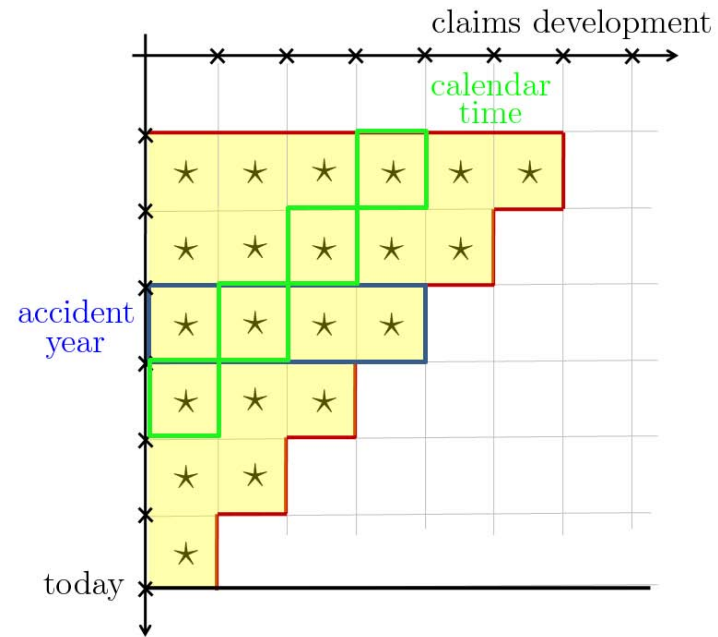
Lexis diagram in insurance

note that whatever the way we look at triangles, there are still three dimensions, year of occurrence or birth, age or development and calendar time,

in life insurance



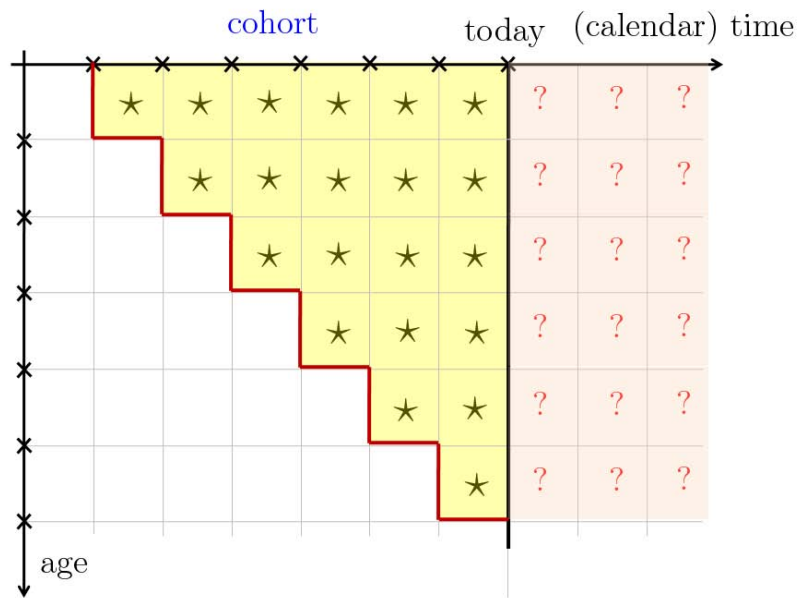
in nonlife insurance



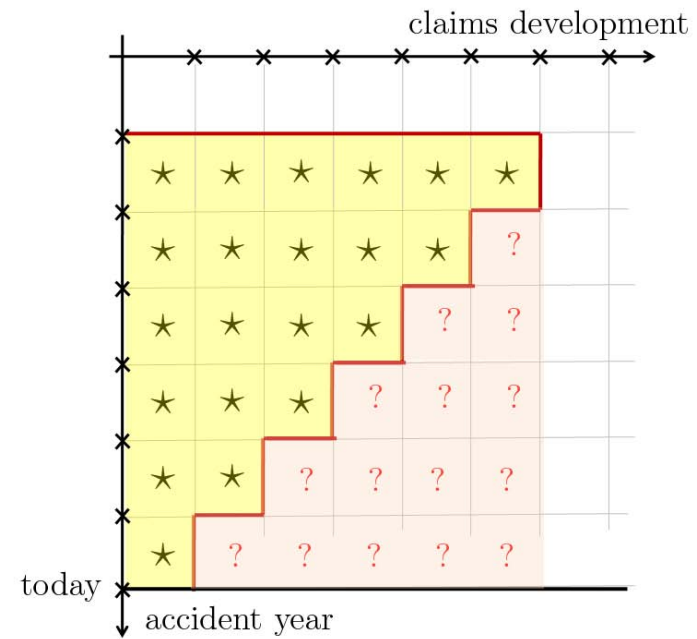
Lexis diagram in insurance

and in both cases, we want to answer a **prediction** question...

in **life** insurance



in **nonlife** insurance



Loss development triangle in nonlife insurance

Let **PAID** denote a triangle of cumulated payments, over time

```
> PAID
```

```
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 3209 4372 4411 4428 4435 4456
[2,] 3367 4659 4696 4720 4730  NA
[3,] 3871 5345 5398 5420  NA  NA
[4,] 4239 5917 6020  NA  NA  NA
[5,] 4929 6794  NA  NA  NA  NA
[6,] 5217  NA  NA  NA  NA  NA
```

- row i the accident year
- column j the development year
- diagonal $i + j$ the calendar year (for incremental payments)
- $Y_{i,j}$ are incremental payments
- $C_{i,j}$ are cumulated payments, per accident year, $C_{i,j} = Y_{i,0} + Y_{i,1} + \cdots + Y_{i,j}$

Incurred But Not Reported - IBNR

At time n , all claims occurred before have not necessarily been (fully) reported, or settled, so claims manager have to predict the total amount of payments. Let \mathcal{H}_n denote the information available at time n (the upper part of the triangle)

$$\mathcal{H}_n = \{(Y_{i,j}), i + j \leq n\} = \{(C_{i,j}), i + j \leq n\}.$$

For accident year i , we want to get a best estimate of the total amount of payment, i.e.

$$\widehat{C}_{i,\infty}^{(n-i)} = \lim_{j \rightarrow \infty} \mathbb{E}[C_{i,j} | \mathcal{H}_n] = \mathbb{E}[C_{i,\infty} | \mathcal{H}_n]$$

and the difference with the payment done as at year i will be the required reserve

$$\widehat{R}_i = \widehat{C}_{i,\infty}^{(n-i)} - C_{i,n-i}.$$

One interesting quantity can be the (ultimate) uncertainty,

$$\text{Var}[C_{i,\infty} | \mathcal{H}_n] \text{ or } \text{Var}[\widehat{C}_{i,\infty}^{(n-i)}]$$

One year uncertainty in Solvency II

In Solvency II, it is required to study the *CDR* (*claims development result*) defined as

$$\Delta_i^n = \widehat{C}_{i,\infty}^{(n-i+1)} - \widehat{C}_{i,\infty}^{(n-i)} = CDR_i(n),$$

Note that $\mathbb{E}[\Delta_i^n | \mathcal{F}_{i,n-i}] = 0$, but insurers have to estimate (and set a reserve for that uncertainty) $\text{Var}[\Delta_i^n | \mathcal{F}_{i,n-i}]$.

Chain Ladder

The most popular model is the **Chain Ladder** methodology, assuming that

$$C_{i,j+1} = \lambda_j \cdot C_{i,j} \text{ for all } i, j = 1, \dots, n.$$

A natural estimator for λ_j is

$$\hat{\lambda}_j = \frac{\sum_{i=1}^{n-j} C_{i,j+1}}{\sum_{i=1}^{n-j} C_{i,j}} \text{ for all } j = 1, \dots, n-1.$$

Then it is natural to predict the non-observed part of the triangle using

$$\hat{C}_{i,j} = \left[\hat{\lambda}_{n+1-i} \cdots \hat{\lambda}_{j-1} \right] \cdot C_{i,n+1-i}.$$

```
> k <- 1
> sum(PAID[1:(n1-k),k+1])/sum(PAID[1:(n1-k),k])
[1] 1.380933
```

Chain Ladder

Those transition factors can be obtained using a simple loop

```
> LAMBDA <- rep(NA,nc-1)
> for(k in 1:(nc-1)){
+ LAMBDA[k]=(sum(PAID[1:(nl-k),k+1])/sum(PAID[1:(nl-k),k]))}
> LAMBDA
[1] 1.380933 1.011433 1.004343 1.001858 1.004735
```

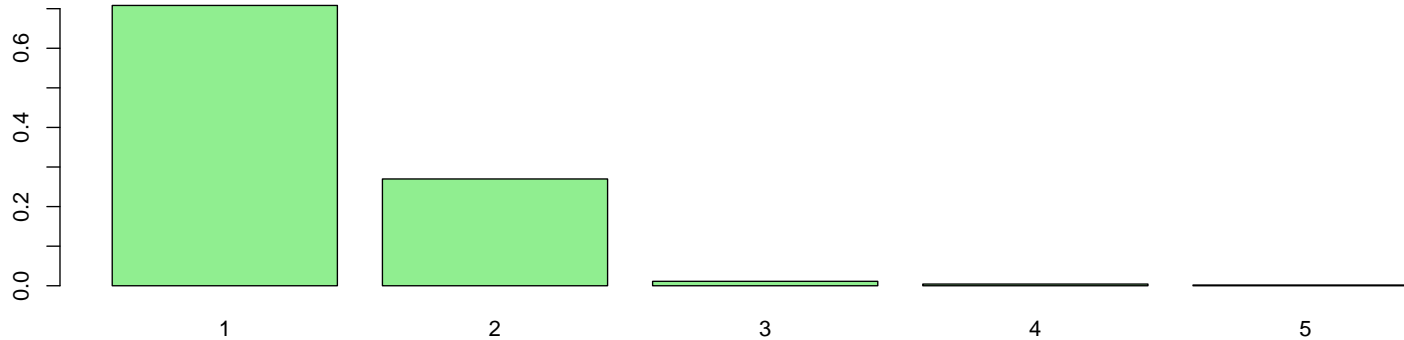
It is possible to rewrite $C_{i,j+1} = \lambda_j \cdot C_{i,j}$ for all $i, j = 1, \dots, n$, as follows

$$C_{i,j} = \gamma_j \cdot C_{i,\infty} \text{ or } Y_{i,j} = \varphi_j \cdot C_{i,\infty}.$$

```
> (GAMMA <- rev(cumprod(rev(1/LAMBDA))))
[1] 0.7081910 0.9779643 0.9891449 0.9934411 0.9952873
> (PHI <- c(GAMMA[1],diff(GAMMA)))
[1] 0.708191033 0.269773306 0.011180591 0.004296183 0.001846141
```

Chain Ladder

```
> barplot(PHI, names=1:5)
```



	0	1	2	3	4	n
λ_j	1,38093	1,01143	1,00434	1,00186	1,00474	1,0000
γ_j	70,819%	97,796%	98,914%	99,344%	99,529%	100,000%
φ_j	70,819%	26,977%	1,118%	0,430%	0,185%	0,000%

Chain Ladder

Note that we can write

$$\hat{\lambda}_j = \sum_{i=1}^{n-j} \omega_{i,j} \cdot \lambda_{i,j} \text{ où } \omega_{i,j} = \frac{C_{i,j}}{\sum_{i=1}^{n-j} C_{i,j}} \text{ et } \lambda_{i,j} = \frac{C_{i,j+1}}{C_{i,j}}.$$

i.e.

```
> k <- 1
> weighted.mean(x=PAID[,k+1]/PAID[,k],w=PAID[,k],na.rm=TRUE)
[1] 1.380933
```

An alternative is to write that weighted mean as the solution of a weighted least squares problem, i.e.

```
> lm(PAID[,k+1]~0+PAID[,k],weights=1/PAID[,k])$coefficients
PAID[, k]
1.380933
```

The main interest of that expression is that it remains valid when triangle contains `NA`'s


```

> LAMBDA <- rep(NA,nc-1)
> for(k in 1:(nc-1)){
+ LAMBDA[k]=lm(PAID[,k+1]~0+PAID[,k],
+ weights=1/PAID[,k])$coefficients}
> LAMBDA
[1] 1.380933 1.011433 1.004343 1.001858 1.004735

```

The idea here is to iterate, by column,

```

> TRIANGLE <- PAID
> for(i in 1:(nc-1)){
+ TRIANGLE[(n1-i+1):(n1),i+1]=LAMBDA[i]*TRIANGLE[(n1-i+1):(n1),i]}
> TRIANGLE
      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 3209 4372.000 4411.000 4428.000 4435.000 4456.000
[2,] 3367 4659.000 4696.000 4720.000 4730.000 4752.397
[3,] 3871 5345.000 5398.000 5420.000 5430.072 5455.784
[4,] 4239 5917.000 6020.000 6046.147 6057.383 6086.065
[5,] 4929 6794.000 6871.672 6901.518 6914.344 6947.084
[6,] 5217 7204.327 7286.691 7318.339 7331.939 7366.656

> tail.factor <- 1

```

```
> chargeultime <- TRIANGLE[,nc]*tail.factor
> paiements <- diag(TRIANGLE[,nc:1])
> (RESERVES <- chargeultime-paiements)
[1] 0.00000 22.39684 35.78388 66.06466 153.08358 2149.65640
```

Here `sum(RESERVES)` is equal to `2426.985`, which is the best estimate of the amount of required reserves,

```
> DIAG <- diag(TRIANGLE[,nc:1])
> PRODUIT <- c(1,rev(LAMBDA))
> sum((cumprod(PRODUIT)-1)*DIAG)
[1] 2426.985
```

It is possible to write a function `Chainladder()` which return the total amount of reserves

```
> Chainladder<-function(TR,f=1){
+ nc <- ncol(TR); nl <- nrow(TR)
+ L <- rep(NA,nc-1)
+ for(k in 1:(nc-1)){
```

```
+ L[k]=lm(TR[,k+1]~0+ TR[,k],
+ weights=1/TR[,k])$coefficients}
+ TRc <- TR
+ for(i in 1:(nc-1)){
+ TRc[(nl-i+1):(nl),i+1]=L[i]* TRc[(nl-i+1):(nl),i]}
+ C <- TRc[,nc]*f
+ R <- (cumprod(c(1,rev(L)))-1)*diag(TR[,nc:1])
+ return(list(charge=C,reserves=R,restot=sum(R)))
+ }
> Chainladder(PAID)
$charge
[1] 4456.000 4752.397 5455.784 6086.065 6947.084 7366.656

$reserves
[1] 0.00000 22.39684 35.78388 66.06466 153.08358 2149.65640

$restot
[1] 2426.985
```

A stochastic model for paiments

Assume is that $(C_{i,j})_{j \geq 0}$ is a Markovian process and that there are $\boldsymbol{\lambda} = (\lambda_j)$ and $\boldsymbol{\sigma} = (\sigma_j^2)$ such that

$$\begin{cases} \mathbb{E}(C_{i,j+1} | \mathcal{H}_{i+j}) = \mathbb{E}(C_{i,j+1} | C_{i,j}) = \lambda_j \cdot C_{i,j} \\ \text{Var}(C_{i,j+1} | \mathcal{H}_{i+j}) = \text{Var}(C_{i,j+1} | C_{i,j}) = \sigma_j^2 \cdot C_{i,j} \end{cases}$$

Under those assumptions,

$$\mathbb{E}(C_{i,j+k} | \mathcal{H}_{i+j}) = \mathbb{E}(C_{i,j+k} | C_{i,j}) = \lambda_j \cdot \lambda_{j+1} \cdots \lambda_{j+k-1} C_{i,j}$$

MACK (1993) assumes moreover that $(C_{i,j})_{j=1,\dots,n}$ and $(C_{i',j})_{j=1,\dots,n}$ are independent for all $i \neq i'$.

Remark : It is also possible to write

$$C_{i,j+1} = \lambda_j C_{i,j} + \sigma_j \sqrt{C_{i,j}} \cdot \varepsilon_{i,j},$$

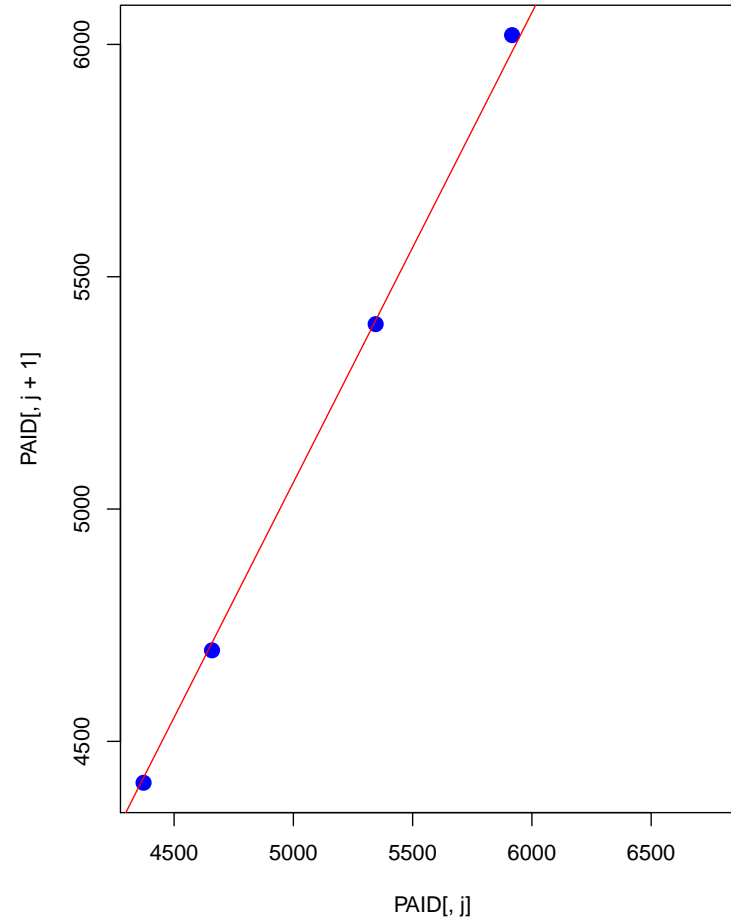
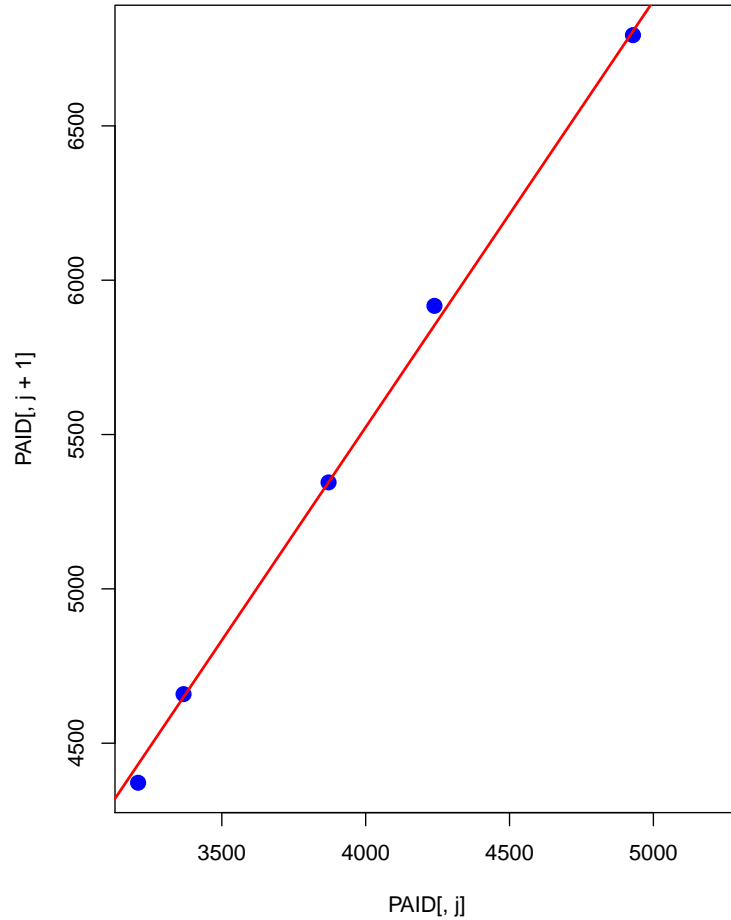
where residuals $(\varepsilon_{i,j})$ are i.i.d., centred, with unit variance.

From that expression, one might think of weighed least squares techniques, to estimation transition factors, i.e. given j , solve

$$\min \left\{ \sum_{i=1}^{n-j} \frac{1}{C_{i,j}} (C_{i,j+1} - \lambda_j C_{i,j})^2 \right\}.$$

Remark : The interpretation of the first assumption is that points $C_{.,j+1}$ versus $C_{.,j}$ should be on a straight line that goes through 0.

```
> par(mfrow = c(1, 2))
> j=1
> plot(PAID[,j],PAID[,j+1],pch=19,cex=1.5)
> abline(lm(PAID[,j+1]~0+PAID[,j],weights=1/PAID[,j]))
> j=2
> plot(PAID[,j],PAID[,j+1],pch=19,cex=1.5)
> abline(lm(PAID[,j+1]~0+PAID[,j],weights=1/PAID[,j]))
> par(mfrow = c(1, 1))
```



Then, a natural idea is to use *weighted residuals*), $\varepsilon_{i,j} = \frac{C_{i,j+1} - \hat{\lambda}_j C_{i,j}}{\sqrt{C_{i,j}}}$.

```
> j=1
> RESIDUALS <- (PAID[,j+1]-LAMBDA[j]*PAID[,j])/sqrt(PAID[,j])
```

Based on those residuals, it is then possible to estimate the volatility parameter

$$\widehat{\sigma}_j^2 = \frac{1}{n-j-1} \sum_{i=0}^{n-j-1} \left(\frac{C_{i,j+1} - \widehat{\lambda}_j C_{i,j}}{\sqrt{C_{i,j}}} \right)^2$$

or equivalently

$$\widehat{\sigma}_j^2 = \frac{1}{n-j-1} \sum_{i=0}^{n-j-1} \left(\frac{C_{i,j+1}}{C_{i,j}} - \widehat{\lambda}_j \right)^2 \cdot C_{i,j}$$

```
> lambda <- PAID[,2:nc]/PAID[,1:(nc-1)]
> SIGMA <- rep(NA,nc-1)
> for(i in 1:(nc-1)){
+ D <- PAID[,i]*(lambda[,i]-t(rep(LAMBDA[i],nc)))^2
+ SIGMA[i] <- 1/(nc-i-1)*sum(D[,1:(nc-i)])}
> SIGMA[nc-1] <- min(SIGMA[(nc-3):(nc-2)])
> (SIGMA=sqrt(SIGMA))
[1] 0.72485777 0.32036422 0.04587297 0.02570564 0.02570564
```

Tail factor, and closure of claim files

Assume that there exists $\lambda_\infty > 1$ such that

$$C_{i,\infty} = C_{i,n} \times \lambda_\infty.$$

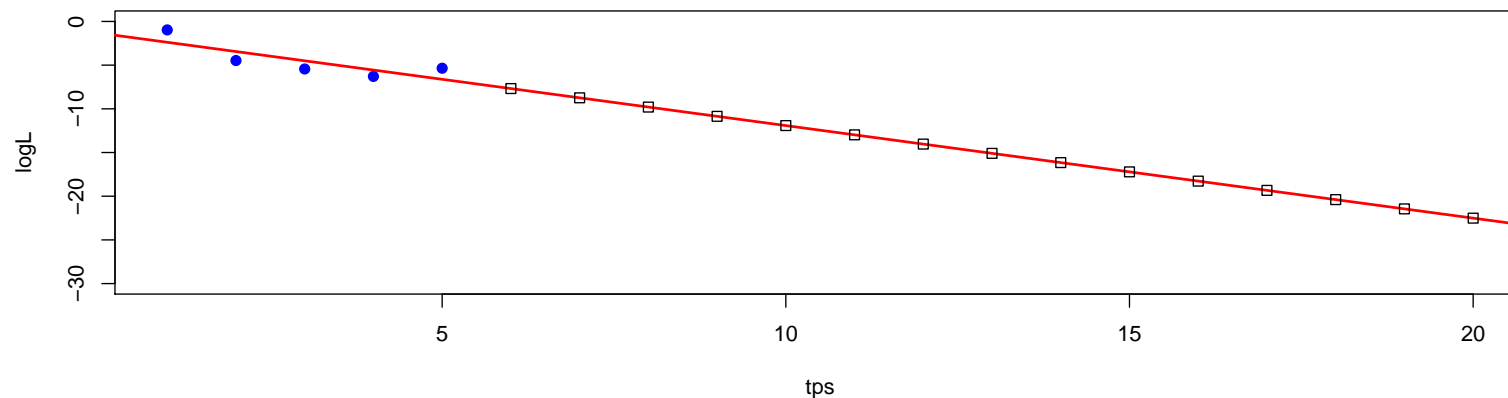
A standard technique to extrapolate λ_i is to consider an exponential extrapolation (i.e. linear extrapolation on $\log(\lambda_k - 1)$'s), then set

$$\lambda_\infty = \prod_{k \geq n} \hat{\lambda}_k.$$

```
> logL <- log(LAMBDA-1)
> tps <- 1:(nc-1)
> modele <- lm(logL~tps)
> plot(tps,logL,xlim=c(1,20),ylim=c(-30,0))
> abline(modele)
> tpsP <- seq(6,1000)
> logP <- predict(modele,newdata=data.frame(tps=tpsP))
> points(tpsP,logP ,pch=0)
> (facteur <- prod(exp(logP)+1))
[1] 1.000707
```


I.e. 0.07% should be added on top of estimated final charge

```
> chargeultime <- TRIANGLE[,nc]*facteur
> paiements <- diag(TRIANGLE[,nc:1])
> (RESERVES <- chargeultime-paiements)
[1] 3.148948 25.755248 39.639346 70.365538 157.992918 2154.862234
> sum(RESERVES)
[1] 2451.764
```



The ChainLadder package

```
> library(ChainLadder)
> MackChainLadder(PAID)
MackChainLadder(Triangle = PAID)
```

	Latest	Dev.To.Date	Ultimate	IBNR	Mack.S.E	CV(IBNR)
1	4,456	1.000	4,456	0.0	0.000	NaN
2	4,730	0.995	4,752	22.4	0.639	0.0285
3	5,420	0.993	5,456	35.8	2.503	0.0699
4	6,020	0.989	6,086	66.1	5.046	0.0764
5	6,794	0.978	6,947	153.1	31.332	0.2047
6	5,217	0.708	7,367	2,149.7	68.449	0.0318

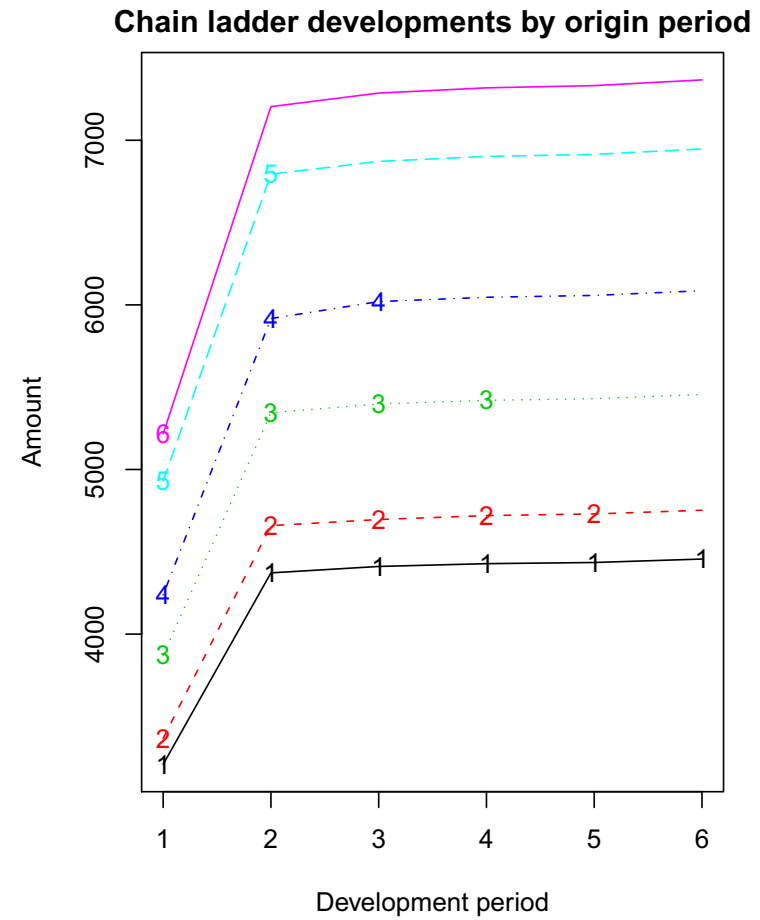
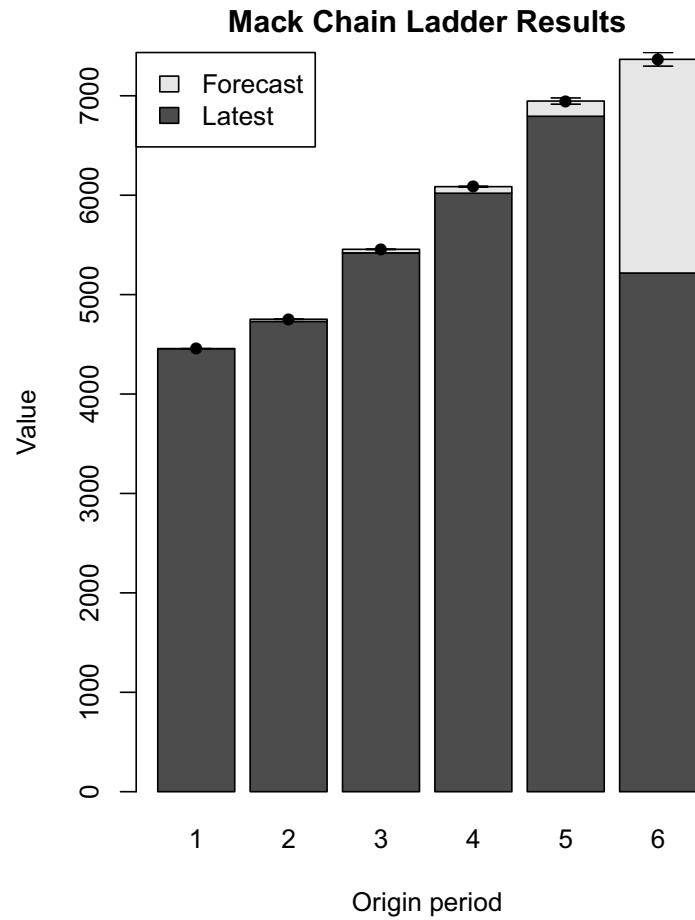
In this first part, we have outputs per accident year, namely reserves \hat{R}_i in the **IBNR**, column, e.g. **2,149.7**, for present year, and $\widehat{\text{mse}}(\hat{R}_i)$ given by **Mack S.E.** e.g. **68.449** for present year.

The ChainLadder package

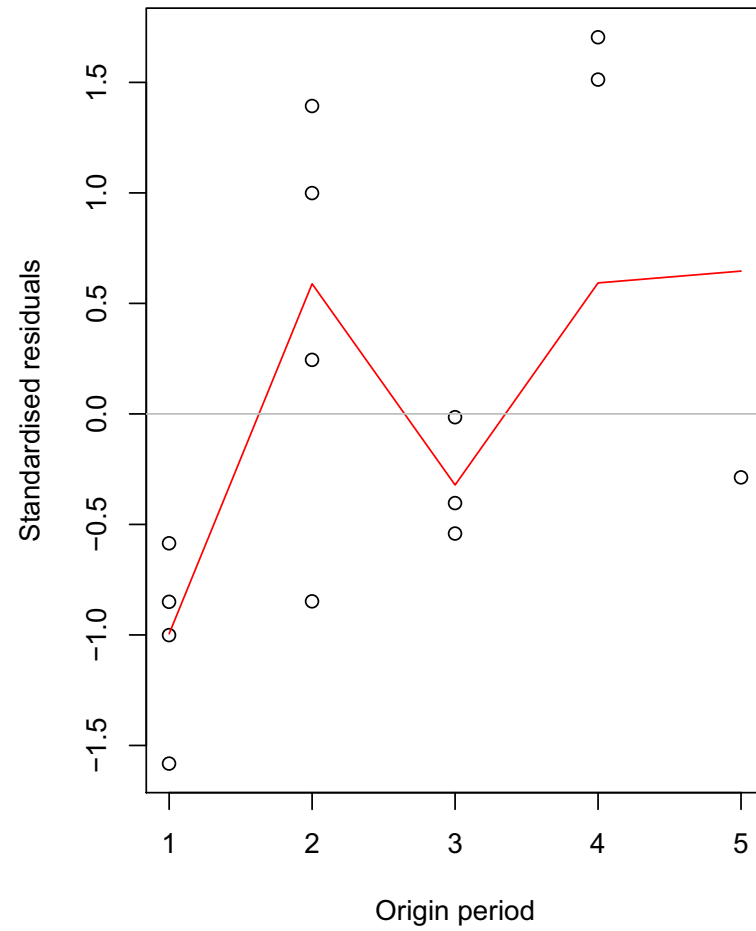
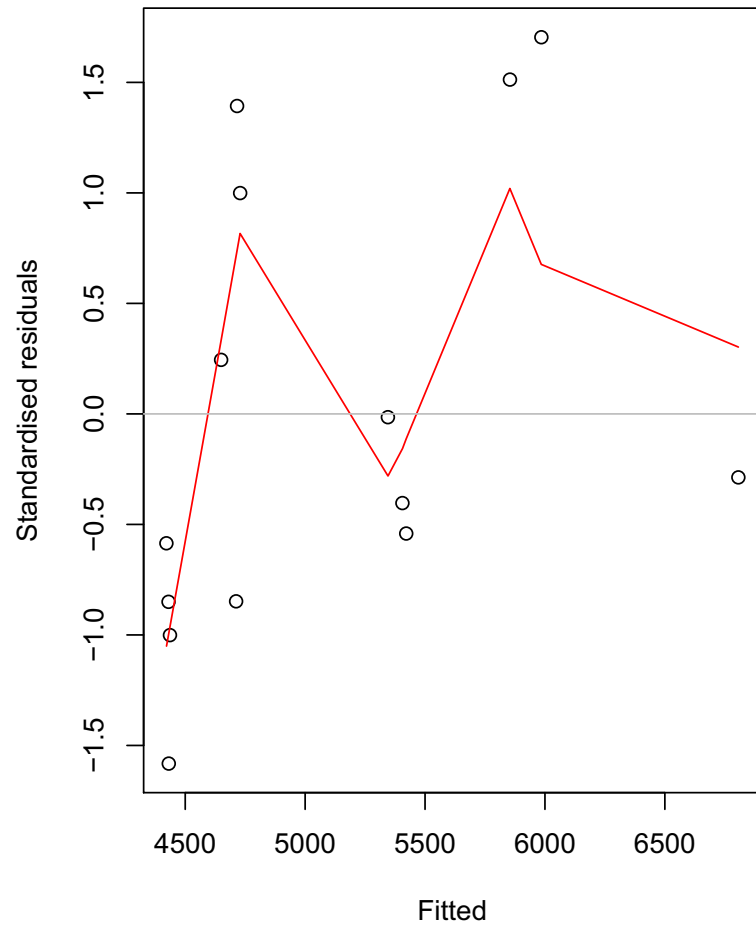
	Totals
Latest:	32,637.00
Ultimate:	35,063.99
IBNR:	2,426.99
Mack S.E.:	79.30
CV(IBNR):	0.03

Here, the total amount of reserves \hat{R} is `IBNR`, qui vaut `2,426.99`, as well as $\widehat{\text{mse}}(\hat{R})$ given by `Mack S.E.` i.e. `79.30`. Function `plot()` can also be used on the output of that function.

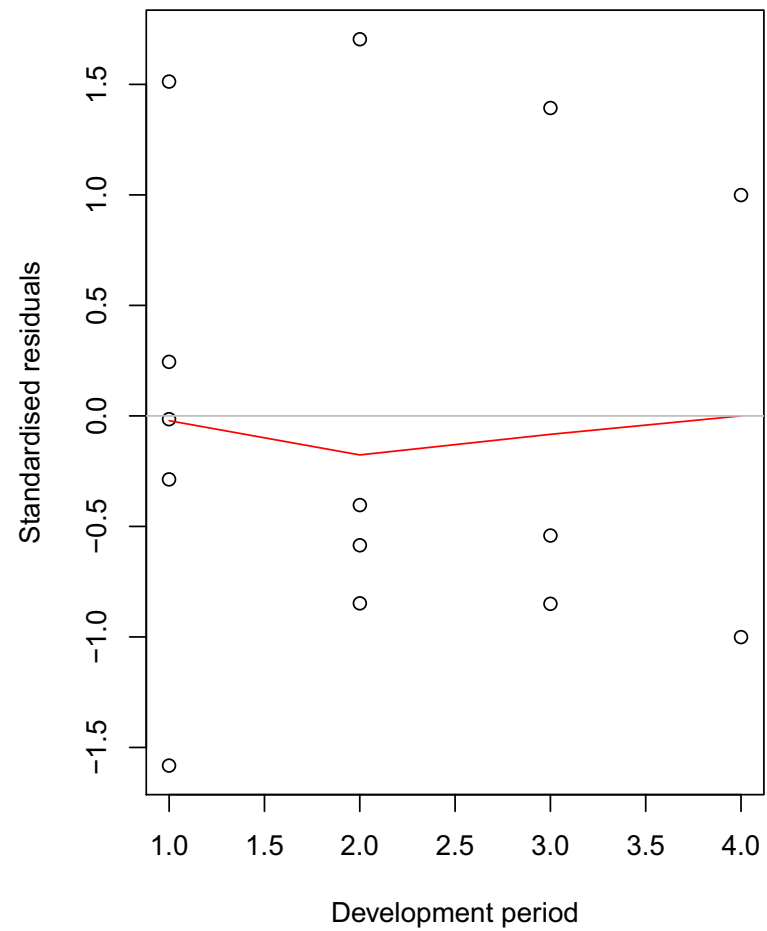
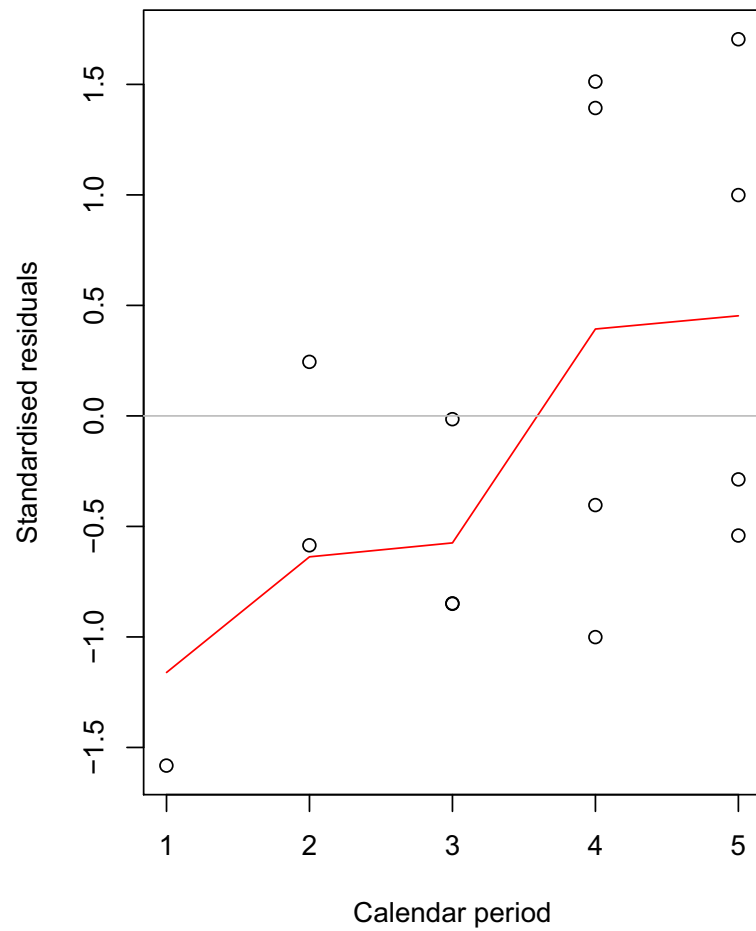
The ChainLadder package



The ChainLadder package



The ChainLadder package



One year uncertainty

MERZ & WÜTHRICH (2008) studied the evolution of $\text{CDR}_i(n)$ with time (n). They proved that

$$\widehat{\text{msepc}}_{n-1}(\text{CDR}_i(n)) = \widehat{C}_{i,\infty}^2 \left(\widehat{\Gamma}_{i,n} + \widehat{\Delta}_{i,n} \right)$$

where

$$\widehat{\Delta}_{i,n} = \frac{\widehat{\sigma}_{n-i+1}^2}{\widehat{\lambda}_{n-i+1}^2 S_{n-i+1}^{n+1}} + \sum_{j=n-i+2}^{n-1} \left(\frac{C_{n-j+1,j}}{S_j^{n+1}} \right)^2 \frac{\widehat{\sigma}_j^2}{\widehat{\lambda}_j^2 S_j^n}$$

and

$$\widehat{\Gamma}_{i,n} = \left(1 + \frac{\widehat{\sigma}_{n-i+1}^2}{\widehat{\lambda}_{n-i+1}^2 C_{i,n-i+1}} \right) \prod_{j=n-i+2}^{n-1} \left(1 + \frac{\widehat{\sigma}_j^2}{\widehat{\lambda}_j^2 [S_j^{n+1}]^2} C_{n-j+1,j} \right) - 1$$

MERZ & WÜTHRICH (2008) mentioned that one can approximate the term above as

$$\widehat{\Gamma}_{i,n} \approx \frac{\widehat{\sigma}_{n-i+1}^2}{\widehat{\lambda}_{n-i+1}^2 C_{i,n-i+1}} + \sum_{j=n-i+2}^{n-1} \left(\frac{C_{n-j+1,j}}{S_j^{n+1}} \right)^2 \frac{\widehat{\sigma}_j^2}{\widehat{\lambda}_j^2 C_{n-j+1,j}}$$

using $\prod(1 + u_i) \approx 1 + \sum u_i$, which is valid if u_i is small i.e.

$$\frac{\widehat{\sigma}_j^2}{\widehat{\lambda}_j^2} \ll C_{n-j+1,j}.$$

It is possible to use function `MackMerzWuthrich()`

```
> MackMerzWuthrich(PAID)
```

	MSEP Mack	MSEP MW app.	MSEP MW ex.
1	0.0000000	0.0000000	0.0000000
2	0.6393379	1.424131	1.315292
3	2.5025153	2.543508	2.543508
4	5.0459004	4.476698	4.476698
5	31.3319292	30.915407	30.915407
6	68.4489667	60.832875	60.832898
tot	79.2954414	72.574735	72.572700

Regression and claims reserving

DE VYLDER (1978) suggested the following model, for incremental payments

$$Y_{i,j} \sim \mathcal{N}(\alpha_i \cdot \beta_j, \sigma^2), \text{ for all } i, j,$$

i.e. an accident year factor α_i and a development factor β_j . It is possible to use least squares techniques to estimate those factors

$$(\hat{\alpha}, \hat{\beta}) = \operatorname{argmin} \left\{ \sum_{i,j} [Y_{i,j} - \alpha_i \cdot \beta_j]^2 \right\}.$$

Here normal equations are

$$\hat{\alpha}_i = \frac{\sum_j Y_{i,j} \cdot \hat{\beta}_j}{\sum_j \hat{\beta}_j^2} \text{ et } \hat{\beta}_j = \frac{\sum_i Y_{i,j} \cdot \hat{\alpha}_i}{\sum_i \hat{\alpha}_i^2},$$

Regression and claims reserving

An alternative is to consider an additive model for the logarithm of incremental payments

$$\log Y_{i,j} \sim \mathcal{N}(a_i + b_j, \sigma^2), \text{ for all } i, j.$$

```
> ligne <- rep(1:nl, each=nc); colonne <- rep(1:nc, nl)
> INC <- PAID
> INC[,2:6] <- PAID[,2:6]-PAID[,1:5]
> Y <- as.vector(INC)
> lig <- as.factor(ligne)
> col <- as.factor(colonne)
> reg <- lm(log(Y)~col+lig)
```

```
> summary(reg)
Call:
lm(formula = log(Y) ~ col + lig)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.9471      0.1101  72.188 6.35e-15 ***
col2           0.1604      0.1109   1.447  0.17849
col3           0.2718      0.1208   2.250  0.04819 *
col4           0.5904      0.1342   4.399  0.00134 **
col5           0.5535      0.1562   3.543  0.00533 **
col6           0.6126      0.2070   2.959  0.01431 *
lig2          -0.9674      0.1109  -8.726 5.46e-06 ***
lig3          -4.2329      0.1208 -35.038 8.50e-12 ***
lig4          -5.0571      0.1342 -37.684 4.13e-12 ***
lig5          -5.9031      0.1562 -37.783 4.02e-12 ***
lig6          -4.9026      0.2070 -23.685 4.08e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1753 on 10 degrees of freedom
(15 observations deleted due to missingness)
```

Multiple R-squared: 0.9975, Adjusted R-squared: 0.9949
 F-statistic: 391.7 on 10 and 10 DF, p-value: 1.338e-11

An unbiased estimator for $\hat{Y}_{i,j}$ is then $\hat{Y}_{i,j} = \exp[\hat{a}_i + \hat{b}_j + \hat{\sigma}^2/2]$, i.e.

```
> sigma <- summary(reg)$sigma
> (INCpred <- matrix(exp(logY+sigma^2/2),nl,nc))
      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 2871.209 1091.278 41.66208 18.27237  7.84125 21.32511
[2,] 3370.826 1281.170 48.91167 21.45193  9.20570 25.03588
[3,] 3767.972 1432.116 54.67438 23.97937 10.29030 27.98557
[4,] 5181.482 1969.357 75.18483 32.97495 14.15059 38.48403
[5,] 4994.082 1898.131 72.46559 31.78233 13.63880 37.09216
[6,] 5297.767 2013.554 76.87216 33.71498 14.46816 39.34771
> sum(exp(logY[is.na(Y)==TRUE]+sigma^2/2))
[1] 2481.857
```

A Poisson Regression on incremental payments

HACHEMEISTER & STANARD (1975), KREMER (1985) and MACK (1991) observed that a log-Poisson regression yield exactly the same amount of reserves as the Chain Ladder technique

```
> CL <- glm(Y~lig+col, family=poisson)
> summary(CL)
```

Call:

```
glm(formula = Y ~ lig + col, family = poisson)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	8.05697	0.01551	519.426	< 2e-16	***
lig2	-0.96513	0.01359	-70.994	< 2e-16	***
lig3	-4.14853	0.06613	-62.729	< 2e-16	***
lig4	-5.10499	0.12632	-40.413	< 2e-16	***
lig5	-5.94962	0.24279	-24.505	< 2e-16	***
lig6	-5.01244	0.21877	-22.912	< 2e-16	***
col2	0.06440	0.02090	3.081	0.00206	**

```
col3      0.20242    0.02025    9.995 < 2e-16 ***
col4      0.31175    0.01980   15.744 < 2e-16 ***
col5      0.44407    0.01933   22.971 < 2e-16 ***
col6      0.50271    0.02079   24.179 < 2e-16 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for poisson family taken to be 1)
```

```
Null deviance: 46695.269  on 20  degrees of freedom
Residual deviance:    30.214  on 10  degrees of freedom
(15 observations deleted due to missingness)
```

```
AIC: 209.52
```

```
Number of Fisher Scoring iterations: 4
```

Here, the prediction is

```
> Ypred <- predict(CL,newdata=data.frame(lig,col),type="response")
> (INCpred <- matrix(Ypred,nl,nc))
      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 3155.699 1202.110 49.82071 19.14379 8.226405 21.00000
[2,] 3365.605 1282.070 53.13460 20.41717 8.773595 22.39684
[3,] 3863.737 1471.825 60.99889 23.43904 10.072147 25.71173
[4,] 4310.096 1641.858 68.04580 26.14685 11.235734 28.68208
[5,] 4919.862 1874.138 77.67250 29.84594 12.825297 32.73985
[6,] 5217.000 1987.327 82.36357 31.64850 13.599887 34.71719
```

which is the same as Chain-Ladder estimate

```
> sum(Ypred[is.na(Y)==TRUE])
[1] 2426.985
```

Oversdispersion and quasiPoisson regression

Note that usually, the Poisson model might be too conservative (in terms of variance) since there might be overdispersion of incremental payments.

Remark : N has an overdispersed Poisson distribution with dispersion parameter ϕ if $X/\phi \sim \mathcal{P}(\lambda/\phi)$, i.e. $\mathbb{E}(X) = \lambda$ while $\text{Var}(X) = \phi\lambda$

```
> CL <- glm(Y~lig+col, family=quasipoisson)
> summary(CL)
```

Call:

```
glm(formula = Y ~ lig + col, family = quasipoisson)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.3426	-0.4996	0.0000	0.2770	3.9355

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	8.05697	0.02769	290.995	< 2e-16 ***


```

lig2      -0.96513    0.02427 -39.772 2.41e-12 ***
lig3      -4.14853    0.11805 -35.142 8.26e-12 ***
lig4      -5.10499    0.22548 -22.641 6.36e-10 ***
lig5      -5.94962    0.43338 -13.728 8.17e-08 ***
lig6      -5.01244    0.39050 -12.836 1.55e-07 ***
col2       0.06440    0.03731   1.726 0.115054
col3       0.20242    0.03615   5.599 0.000228 ***
col4       0.31175    0.03535   8.820 4.96e-06 ***
col5       0.44407    0.03451  12.869 1.51e-07 ***
col6       0.50271    0.03711  13.546 9.28e-08 ***

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 3.18623)

Null deviance: 46695.269 on 20 degrees of freedom
Residual deviance: 30.214 on 10 degrees of freedom
(15 observations deleted due to missingness)

AIC: NA

Number of Fisher Scoring iterations: 4

Quantifying uncertainty

Recall that in this GLM framework

$$\mathbb{E}(Y_{i,j} | \mathcal{F}_n) = \mu_{i,j} = \exp[\eta_{i,j}]$$

i.e.

$$\hat{Y}_{i,j} = \hat{\mu}_{i,j} = \exp[\hat{\eta}_{i,j}].$$

Using the *delta method* we can write

$$\text{Var}(\hat{Y}_{i,j}) \approx \left| \frac{\partial \mu_{i,j}}{\partial \eta_{i,j}} \right|^2 \cdot \text{Var}(\hat{\eta}_{i,j}),$$

which becomes, with a log link function

$$\frac{\partial \mu_{i,j}}{\partial \eta_{i,j}} = \mu_{i,j}$$

Quantifying uncertainty

In the case of an overdispersed Poisson model (as in [RENSHAW \(1992\)](#)),

$$\mathbb{E} \left([Y_{i,j} - \widehat{Y}_{i,j}]^2 \right) \approx \widehat{\phi} \cdot \widehat{\mu}_{i,j} + \widehat{\mu}_{i,j}^2 \cdot \widehat{\text{Var}}(\widehat{\eta}_{i,j})$$

for the lower part of the triangle. Further,

$$\text{Cov}(\widehat{Y}_{i,j}, \widehat{Y}_{k,l}) \approx \widehat{\mu}_{i,j} \cdot \widehat{\mu}_{k,l} \cdot \widehat{\text{Cov}}(\widehat{\eta}_{i,j}, \widehat{\eta}_{k,l}).$$

Since $\widehat{R} = \sum_{i+j>n} \widehat{Y}_{i,j}$, then

$$\mathbb{E} \left([R - \widehat{R}]^2 \right) \approx \left(\sum_{i+j>n} \widehat{\phi} \cdot \widehat{\mu}_{i,j} \right) + \widehat{\boldsymbol{\mu}}_F' \cdot \widehat{\text{Var}}(\widehat{\boldsymbol{\eta}}_F) \cdot \widehat{\boldsymbol{\mu}}_F,$$

where $\widehat{\boldsymbol{\mu}}_F$ and $\widehat{\boldsymbol{\eta}}_F$ are restrictions of $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\eta}}$ to indices $i + j > n$ (i.e. lower part of the triangle).

Quantifying uncertainty

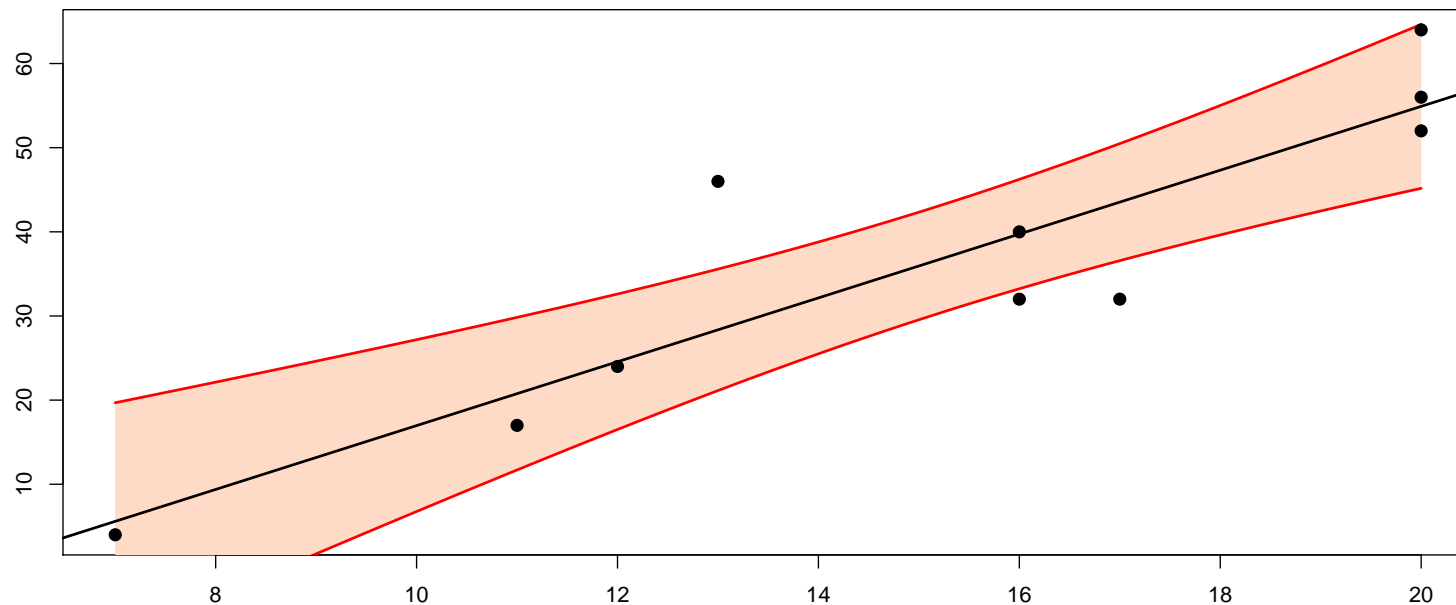
Remark : This expression is only asymptotic.

```
> p <- nl+nc-1;
> phi <- sum(residuals(CL,"pearson")^2)/(sum(is.na(Y)==FALSE)-p)
> Sig <- vcov(CL)
> X <- model.matrix(glm(Ypred~lig+col, family=quasipoisson))
> Cov.eta <- X%%Sig%%t(X)
> Ypred <-predict(CL,newdata=data.frame(lig,col),type="response")*(is.na(Y)==TRUE)
> se2 <- phi * sum(Ypred) + t(Ypred) %% Cov.eta %% Ypred
> sqrt(se2)
      [,1]
[1,] 131.7726
```

Uncertainty and errors in regression models

In regression models, there are two sources in regression models

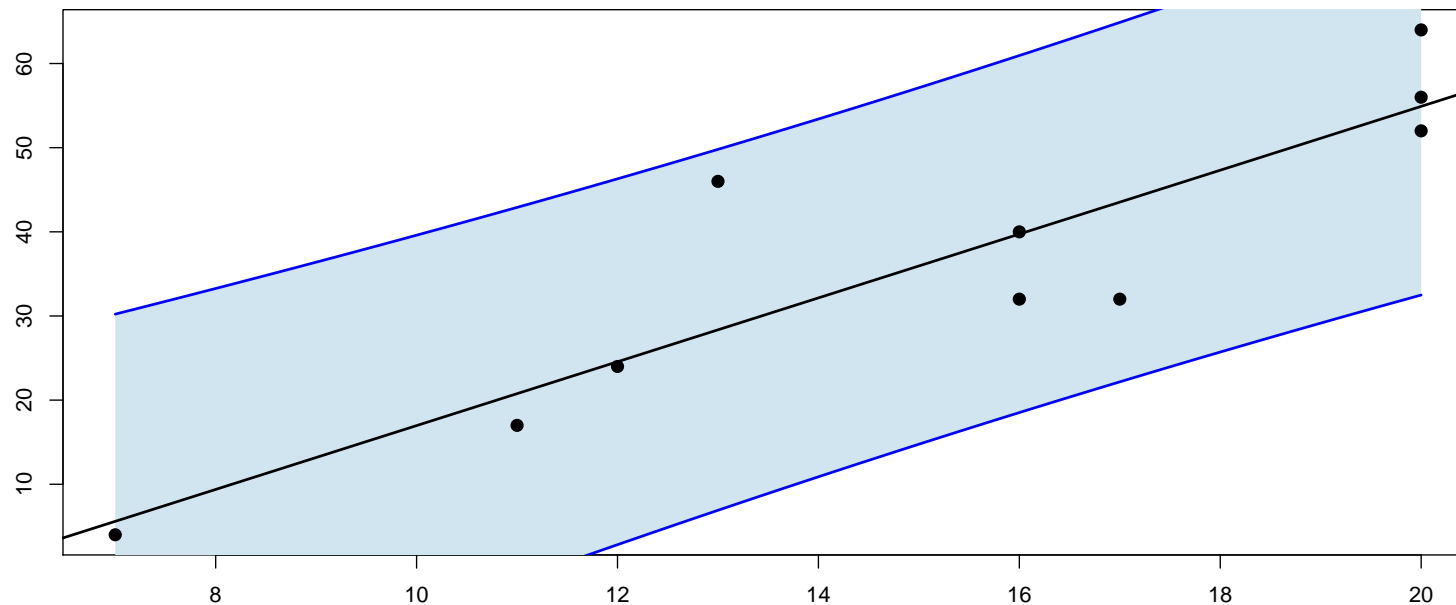
- the **process error** (due to statistical inference) : error on \hat{Y}
- the **variance error** (due to the model approximation) : error on $Y = \hat{Y} + \varepsilon$



Uncertainty and errors in regression models

In regression models, there are two sources in regression models

- the **process error** (due to statistical inference) : error on \hat{Y}
- the **variance error** (due to the model approximation) : error on $Y = \hat{Y} + \varepsilon$



Quantifying uncertainty using simulations

In order to estimate those quantities, consider Person's residuals

$$\hat{\varepsilon}_{i,j} = \frac{Y_{i,j} - \hat{Y}_{i,j}}{\sqrt{\hat{Y}_{i,j}}}.$$

or, in order to have unit variance,

$$\tilde{\varepsilon}_{i,j} = \sqrt{\frac{n}{n-k}} \cdot \frac{Y_{i,j} - \hat{Y}_{i,j}}{\sqrt{\hat{Y}_{i,j}}},$$

where k is the number of parameters in the model (i.e. $2n - 1$).

Quantifying uncertainty using simulations

```
> (residus=residuals(CL,type="pearson"))
```

1	2	3	4	5	6	7	8
9.49e-01	2.40e-02	1.17e-01	-1.08e+00	1.30e-01	-1.01e-13	-1.13e+00	2.77e-01
9	10	11	13	14	15	16	19
5.67e-02	8.92e-01	-2.11e-01	-1.53e+00	-2.21e+00	-1.02e+00	4.24e+00	-4.90e-01
20	21	25	26	31			
7.93e-01	-2.97e-01	-4.28e-01	4.14e-01	-6.20e-15			

```
> n=sum(is.na(Y)==FALSE)
```

```
> k=ncol(PAID)+nrow(PAID)-1
```

```
> (residus=sqrt(n/(n-k))*residus)
```

1	2	3	4	5	6	7	8
1.37e+00	3.49e-02	1.69e-01	-1.57e+00	1.89e-01	-1.46e-13	-1.63e+00	4.02e-01
9	10	11	13	14	15	16	19
8.22e-02	1.29e+00	-3.06e-01	-2.22e+00	-3.21e+00	-1.48e+00	6.14e+00	-7.10e-01
20	21	25	26	31			
1.15e+00	-4.31e-01	-6.20e-01	6.00e-01	-8.99e-15			

Quantifying uncertainty using simulations

```
> epsilon <- rep(NA,nl*nc)
> epsilon[is.na(Y)==FALSE]=residus
> matrix(epsilon,nl,nc)
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 1.37e+00 -1.6346 -2.22 -0.710 -0.62 -8.99e-15
[2,] 3.49e-02 0.4019 -3.21 1.149 0.60 NA
[3,] 1.69e-01 0.0822 -1.48 -0.431 NA NA
[4,] -1.57e+00 1.2926 6.14 NA NA NA
[5,] 1.89e-01 -0.3059 NA NA NA NA
[6,] -1.46e-13 NA NA NA NA NA
```

Generating a pseudo triangle of payments

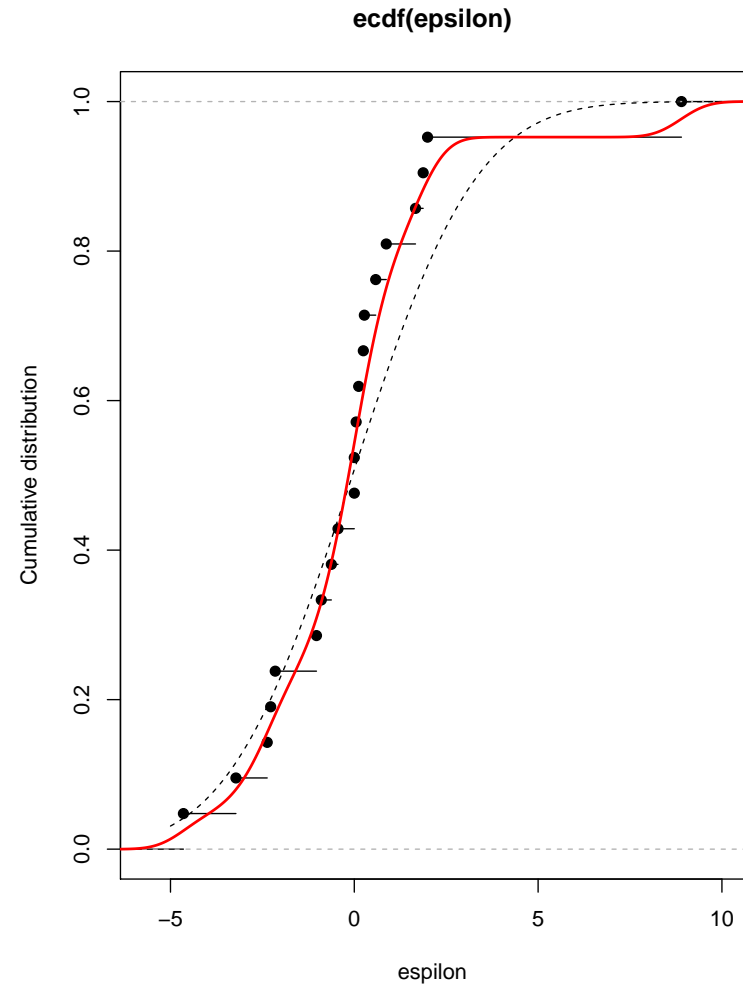
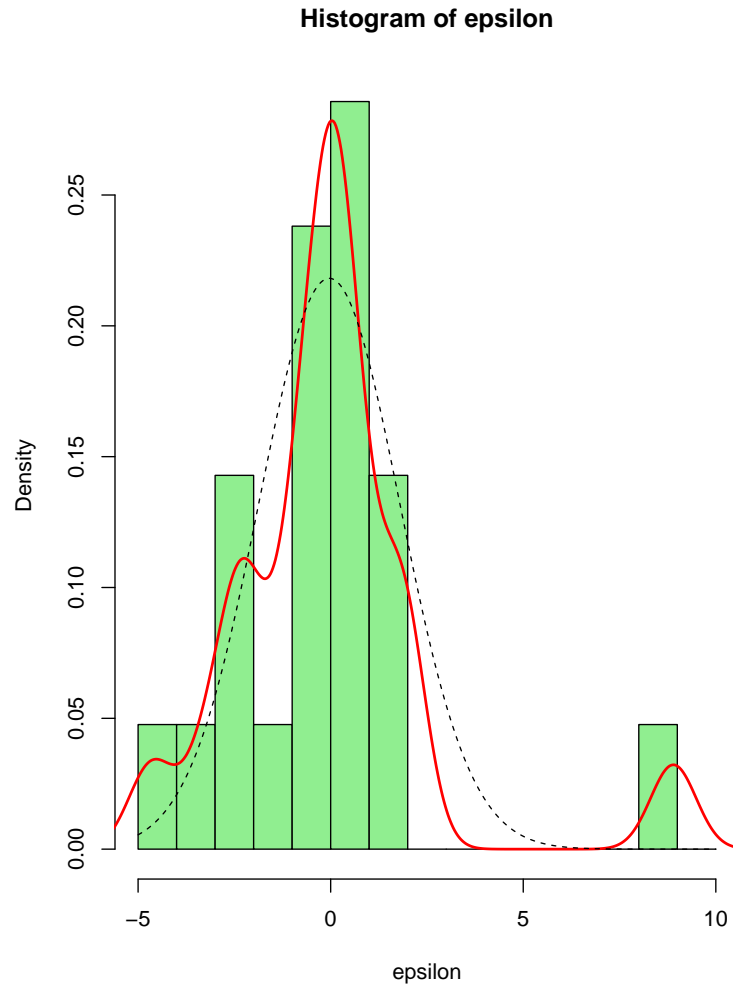
It is then natural to generate errors $\tilde{\varepsilon}^b = (\tilde{\varepsilon}_{i,j}^b)$, to generate a pseudo triangle of payments

$$Y_{i,j}^b = \hat{Y}_{i,j} + \sqrt{\hat{Y}_{i,j}} \cdot \tilde{\varepsilon}_{i,j}^b.$$

```
> par(mfrow = c(1, 2))
> hist(residus,breaks=seq(-3.5,6.5,by=.5),col="grey",proba=TRUE)
> u <- seq(-4,5,by=.01)
> densite <- density(residus)
> lines(densite,lwd=2)
> lines(u,dnorm(u,mean(residus),sd(residus)),lty=2)
> plot(ecdf(residus),xlab="residus",ylab="Fn(residus)")
> lines(u,pnorm(u,mean(residus),sd(residus)),lty=2)
> Femp <- cumsum(densite$y)/sum(densite$y)
> lines(densite$x,Femp,lwd=2)
> par(mfrow = c(1, 1))
```

Bootstrap or parametric model for $\tilde{\varepsilon}$'s?

Generating a pseudo triangle of payments



Generating a pseudo triangle of payments

Once triangle $\mathbf{Y}^b = (Y_{i,j}^b)$ is generate, compute \hat{R}^b , e.g. using Chain Ladder technique. Variance of \hat{R}^b will denote the estimation error.

From triangle $Y_{i,j}^b$, prediction for the lower part of the triangle can be derived $\hat{Y}_{i,j}^b$. In order to generate scenarios of future payments, generate Poisson variates with mean $\hat{Y}_{i,j}^b$,

```
> CLsimul1<-function(triangle){
+   rpoisson(length(triangle),lambda=triangle)}
```

But generating Poisson variates might be too conservative,

```
> (summary(CL)$dispersion)
[1] 3.19
```

Generating quasi Poisson variates

First idea : a Gamma approximation. Recall that $\mathbb{E}(Y) = \lambda$ et $\text{Var}(Y) = \phi\lambda$ for $Y \sim \mathcal{G}(\alpha, \beta)$ i.e $\alpha\beta = \lambda$ and $\alpha\beta^2 = \phi\lambda$.

```
> rqpoisG <- function(n, lambda, phi, roundvalue = TRUE) {
+ b <- phi ; a <- lambda/phi ; r <- rgamma(n, shape = a, scale = b)
+ if(roundvalue){r<-round(r)}
+ return(r)}
```

Second idea : a Negative Binomial variate, with mean μ and variance $\mu + \mu^2/k$, i.e. $\mu = \lambda$ and $k = \lambda(\phi\lambda - 1)^{-1}$, i.e.

```
> rqpoisBN = function(n, lambda, phi) {
+ mu <- lambda ; k <- mu/(phi * mu - 1) ; r <- rnbinom(n, mu = mu, size = k)
+ return(r)}
```

Then

```
> CLsimul2<-function(triangle,surdispersion){
+   rqpoissonG(length(triangle),lambda=triangle,phi=surdispersion)}
```

Quantifying uncertainty using simulations

An alternative to generate future payments is use [MACK \(1993\)](#)'s approach, as suggested in [ENGLAND & VERRALL \(1998\)](#).

From $\hat{\lambda}_j$ and $\hat{\sigma}_j^2$ use the following dynamics

$$C_{i,j+1}^b | C_{i,j}^b \sim \mathcal{N}(\hat{\lambda}_j C_{i,j}^b, \hat{\sigma}_j^2 C_{i,j}^b).$$

Thus, from a triangle of cumulated payments `triangle` a vector of transition values `l` and a vector of conditional variances `s` define

```
> CLsimul3<-function(triangle,l,s){
+ m<-nrow(triangle)
+ for(i in 2:m){
+   triangle[(m-i+2):m,i]<-rnorm(i-1,
+     mean=triangle[(m-i+2):m,i-1]*l[i-1],
+     sd=sqrt(triangle[(m-i+2):m,i-1])*s[i-1])
+ }
+ return(triangle) }
```

Then, we run several simulations to generate best estimates and possible scenarios of payments

```
> ns<-20000
> set.seed(1)
> Yp <- predict(CL,type="response",newdata=base)
> Rs <- R <- rep(NA,ns)
> for(s in 1:ns){
+   serreur <- sample(residus,size=nl*nc,replace=TRUE)
+   E <- matrix(serreur,nl,nc)
+   sY <- matrix(Yp,6,6)+E*sqrt(matrix(Yp,6,6))
+   if(min(sY[is.na(Y)==FALSE])>=0){
+     sbase <- data.frame(sY=as.vector(sY),lig,col)
+     sbase$sY[is.na(Y)==TRUE]=NA
+     sreg <- glm(sY~lig+col,
+ data=sbase,family=poisson(link="log"))
+     sYp <- predict(sreg,type="response",newdata=sbase)
+     R[s] <- sum(sYp[is.na(Y)==TRUE])
+     sYpscenario <- rqpoisG(36,sYp,phi=3.18623)
+     Rs[s] <- sum(sYpscenario[is.na(Y)==TRUE])
+   }}
+ }}
```

Remark : In those simulations, it might be possible to generate negative increments

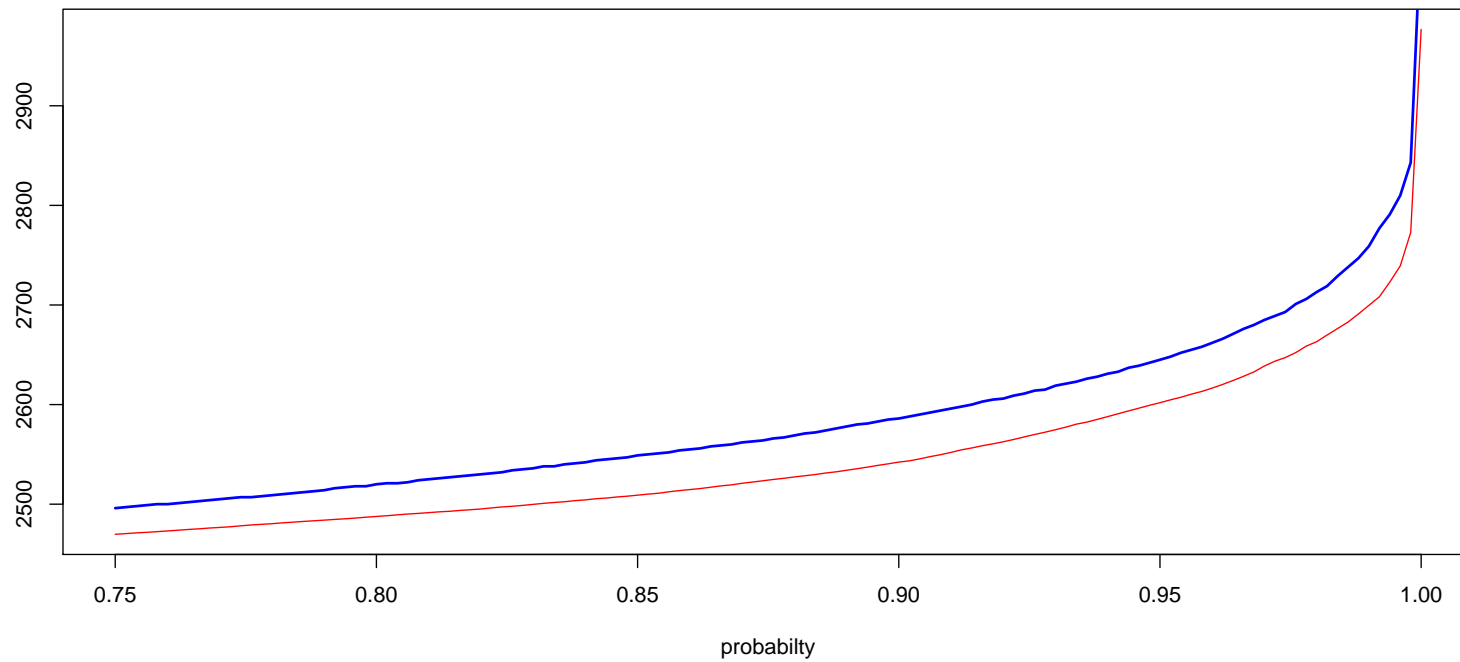
```
> sort(residus)[1:2]
  14   13
-3.21 -2.22
> sort(sqrt(Yp[is.na(Y)==FALSE]))[1:4]
  25   26   19   20
2.87 2.96 4.38 4.52
```

But we might assume that those scenarios while yield low reserves (and thus will not impact high losses quantiles)

```
> Rna <- R
> Rna[is.na(R)==TRUE] <- 0
> Rsna <- Rs
> Rsna[is.na(Rs)==TRUE] <- 0
> quantile(Rna, c(.75, .95, .99, .995))
 75%   95%   99% 99.5%
2470  2602  2700  2729
```



```
> quantile(Rsna,c(.75,.95,.99,.995))  
 75%   95%   99% 99.5%  
2496 2645 2759 2800
```

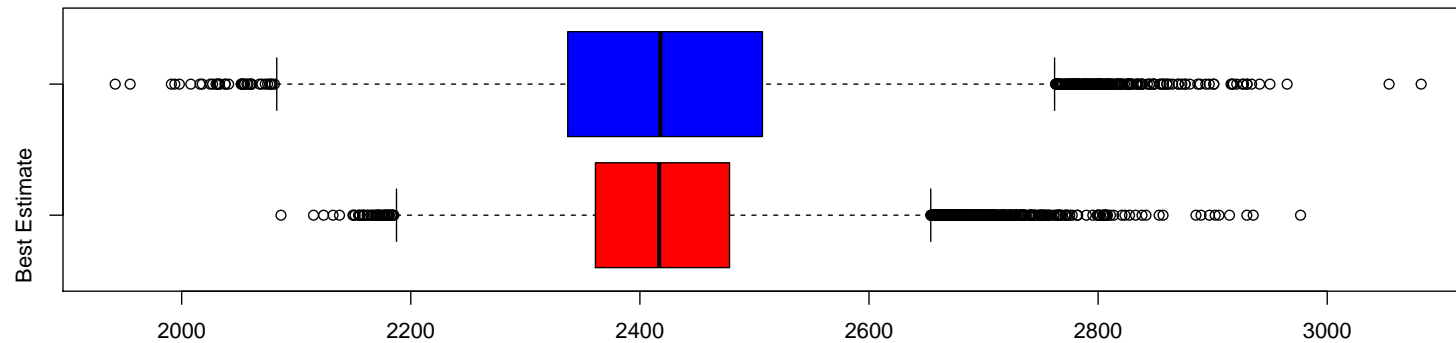


Vector `R` contains best scenarios, while vector `Rs` contains scenarios of payments. In order to visualize their distribution, we have to remove `NA` values

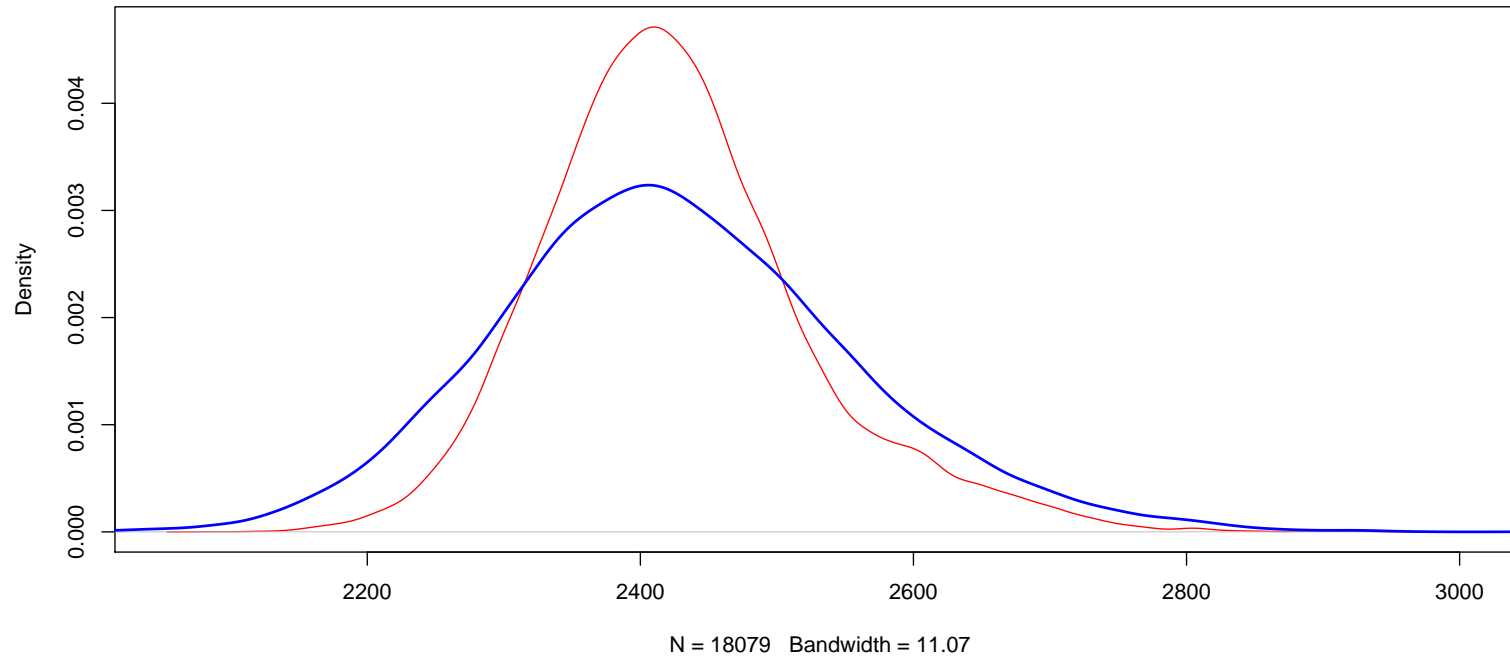
```
> Rnarm <- R[is.na(R)==FALSE]
> Rsnarm <- Rs[is.na(Rs)==FALSE]
```

Here quantiles are slightly higher

```
> quantile(Rnarm,c(.75,.95,.99,.995))
 75%   95%   99% 99.5%
2478  2609  2704  2733
> quantile(Rsnarm,c(.75,.95,.99,.995))
 75%   95%   99% 99.5%
2507  2653  2764  2805
```



```
> plot(density(Rnarm), col="grey", main="")
> lines(density(Rsnarm), lwd=2)
> boxplot(cbind(Rnarm, Rsnarm),
+ col=c("grey", "black"), horizontal=TRUE)
```



Those quantities are quite similar to the one obtained using `bootChainLadder` from `library(ChainLadder)`,

```
> BootChainLadder(PAID,20000,"od.pois")
```

```
BootChainLadder(Triangle = PAID, R = 20000, process.distr = "od.pois")
```

	Latest	Mean Ultimate	Mean IBNR	SD IBNR	IBNR 75%	IBNR 95%
1	4,456	4,456	0.0	0.0	0	0
2	4,730	4,752	22.1	12.0	28	44
3	5,420	5,455	35.3	15.3	44	63
4	6,020	6,085	65.5	19.8	77	101
5	6,794	6,946	152.4	28.6	170	203
6	5,217	7,363	2,146.2	111.6	2,216	2,337

Totals

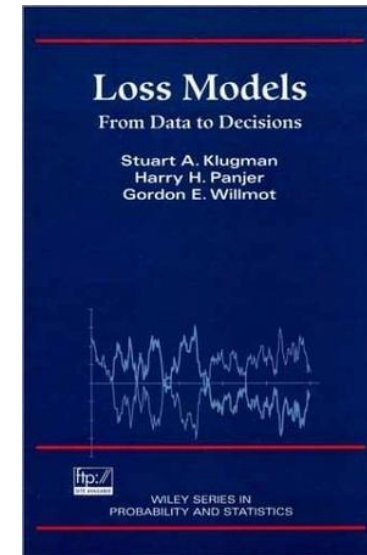
Latest:	32,637
Mean Ultimate:	35,059
Mean IBNR:	2,422
SD IBNR:	132
Total IBNR 75%:	2,504
Total IBNR 95%:	2,651

To go further, loss distributions and credibility

Krugman, S.A., Panjer, H.H. & Willmot, G.E. (1998)

Loss Models

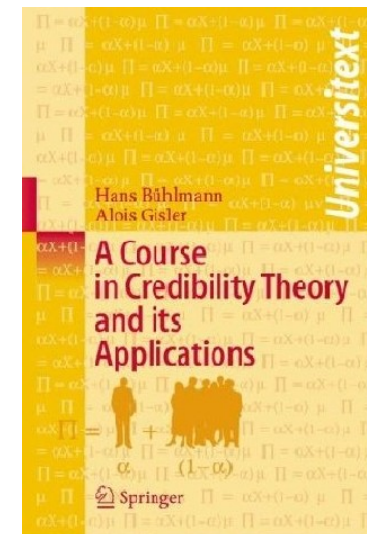
Wiley Series in Probability



Bühlmann, H. & Gisler, A. (2005)

A Course in Credibility Theory and its Applications

Springer Verlag

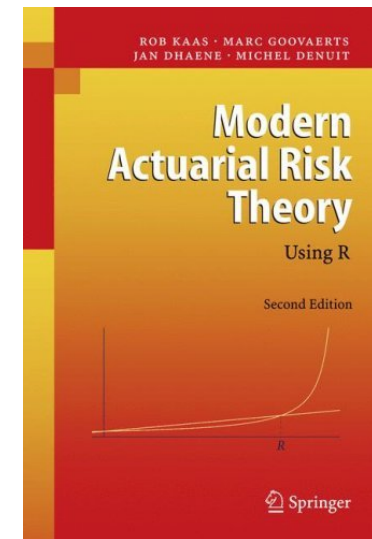


To go further, regression GLMs

Kaas, R., Goovaerts, M, Dhaene, J & Denuit, M. (2009)

Modern Actuarial Risk Theory

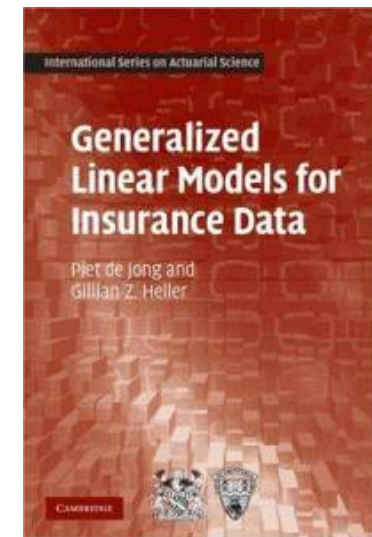
Springer Verlag



de Jong, P. & Heller, G.Z. (2008)

Generalized Linear Models for Insurance Data

Cambridge University Press

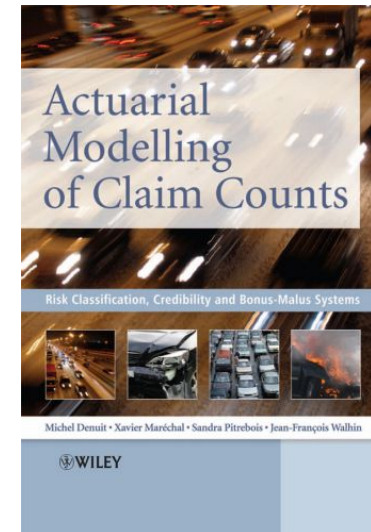


To go further, ratemaking

Denuit, M., Marechal, X., Pitrebois, S.
& Walhin, J.F. (2007)

*Actuarial Modelling of Claim Counts :
Risk Classification, Credibility & Bonus-Malus Systems*

John Wiley & Sons



Ohlsson E. & Johansson, B. (2010)

*Non-Life Insurance Pricing with
Generalized Linear Models*

Springer Verlag

